IMPROVING CUSTOMER RELATIONS

Lars Mathiassen, Gro Bjerknes & Carsten Kristensen

When an organization pursues a systematic strategy to mature its software operation it has severe implications for its customers or clients. Improving requirements management practices is, for example, impossible without changing software contracts and without affecting the interaction between software teams and their customers and future users. Systematic Software Engineering (SSE) therefore decided to initiate activities as part of their SPI program with the specific goal of engaging their customers in improving mutual relationships. Doing so SSE hoped to supply their SPI program with new insights and additional energy, they hoped to develop services that could help their customers mature, and they firmly believed that such efforts would lead to innovations that could help SSE further develop its own processes.

The strategy was simple. SSE wanted to actively engage their customers in the improvement program still with SSE in the leading role. As a first step we organized a one-day workshop with key representatives of the customers. To prepare for the workshop we started to research maturity models that address customer relations and we studied available state-of-the-art knowledge addressing the customer-supplier relationship in software development. In the following, we present useful knowledge about the customer-supplier relationship, we report from our joint workshop with the customers, and we describe SSE’s ongoing efforts to mature their customer relations.

Understanding Customer Relations

Most SPI initiatives operate within a software organization. They concentrate on improving development processes in a rather narrow sense as seen from the point of view of the supplier, i.e. from a project is formed, based on some kind of contractual arrangement with a customer, until a computer system is delivered to the customer. This focus on internal processes has helped many software organizations build project management disciplines and supportive organizational environments that improve their software process capabilities.

There are, however, clear limits to how much a software organization can change its processes without seriously affecting its customer relations. When software organizations mature they therefore need to broaden their perspective on SPI to include customer relations and viewpoints. Seen from the customer organization, projects do not start with explicit contractual arrangements. They are the result of unstructured, problematic situations where the involved actors have different conceptions of both problems and solutions. Neither do projects terminate when the system is delivered. They include the integration of the system into the customer organization’s practices. From such a broader point of view a set of customer centered improvement questions emerge. What are the key elements in understanding systems development as collaborative efforts between customer and supplier organizations? Which opportunities and challenges emerge for buyers of systems development services as supplier
organizations mature? What kinds of practices are needed in customer organizations to participate in and help actively develop mature customer-supplier relationships?

The situations we want to address includes a customer collaborating with a supplier through a development project, i.e. we exclude situations where a customer simply buys commercial off-the-shelf products. The customer needs a new or modified system to pursue one or several business goals and to achieve some desired changes. The customer is typically an organization consisting of many heterogeneous groups. Large organizations may have a purchasing department that is responsible for the solicitation of the system, even though a totally different part of the organization will be system owner and use the system. Smaller organizations will typically establish a project for the procurement of a new system, and the system owner will participate from the beginning. Sometimes, the customer organization lacks the expertise of acquiring systems, and they will buy such expertise, e.g. from a consulting company.

The supplier may be a software house that produces off-the-shelf products for a large market, it may sell software systems that require adaptation, or it may develop software systems on the basis of specific contracts. In the last case, the supplier may be an in-house information technology (IT) department, or a software house. The software house may relate mainly to one customer (typically out-sourced IT departments), or to many customers.

Figure 1 illustrates different roles and interests related to projects between customers and suppliers. The situation is in many cases even more complicated. This complexity of relations makes it difficult to establish well-functioning collaborations. The supplier can, for example, often not interact directly with the future users, but is instead required to interact with a purchasing department, i.e. the buyer.

Further complexity is added because the relationship is enacted at several levels as illustrated in Figure 2. First, there is a constituting level of interactions over time and across different projects. This level is normally enacted by middle management of the customer organization and sales and marketing persons from the supplier organization. There may also be contact between top management on both sides. Marketing and sales people in collaboration with a project leader from the supplier normally negotiate
specific contracts with middle management and the project owner from the customer. The customer may buy consultants for doing parts of this job on their behalf. When a project is running based on a contract we can distinguish two additional levels. At the contractual level the contract tracking and maintenance is done by a steering committee with help of project leaders or managers from both sides. The project is at the development level carried out by project teams on both sides.

**Figure 2.** Customer-supplier relationship levels

The important milestones of each project are at the contractual level: the contract is signed, the system is accepted, and the guarantee period is over. Looking at the development level, however, we find other important milestones: the problem is acknowledged, the requirements are specified, the system is accepted, the system is in use. Thus a project typically lasts longer for the customer than for the supplier. It starts when the customer identifies a problem to solve, and it ends when the solution is implemented as an integral part of the customer’s operation. This basic structure of a project is illustrated in Figure 3 below. We use the term ‘IT project’ rather than the more narrow ‘software project’ to stress this broader perspective.

**Figure 3.** Structure of a project

In the following, we introduce pairs of contrasting concepts that help us understand the nature of the relationship at each of the three levels presented in Figure 2: collaboration and competition at the constituting level; trust and control at the contractual level; and care and engineering at the development level. A well-functioning customer-supplier relationship requires that the involved actors constantly seek to balance these opposites at each level.
Collaboration and Competition

The customer-supplier relation fundamentally includes a mix of three quite different types of contractual arrangements: teams with close collaboration, bureaucratic organizational forms, and market relationships (Williamson [12], Ouchi [8], and Ciborra [1]). The team, in the one extreme, requires a high level of goal congruence between the involved actors. But the performance measure is typically implicit and based on mutual trust and shared interests. In the other extreme, the market is based on explicit measures of performance, i.e. the price. In this case there is no assumption about any form of goal congruence as the basic form of relation on the market is competition. The bureaucracy is an intermediary form in which there is neither a high level of goal congruence nor explicit measures of performance. In this case actors are submitted to work for the goals of an organization through the employment contract.

The customer-supplier relation in IT development can at the constituting level be viewed as a dynamic network of exchanges that are governed by these three ideal types of contractual arrangements. The customer buys services from the supplier and the basic relation is the market. But all sorts of mixtures of market and bureaucratic relationships exist depending on the mutual dependencies between the customer and the supplier organization. Specific development efforts are organized as projects in which the team is the ideal form of collaboration. But such teams are always embedded into the bureaucratic arrangements within the customer and the supplier organizations. The constituting level can hence be understood as a dynamic relationship that balances the intention and need to collaborate with inherent competing interests. Mastenbroek [6] views this kind of relationship as networks of interdependent groups that are governed by collaboration and competition. The involved actors and organizational units are dependent on each other, they have their own interests, and they need to form what Mastenbroek [6] calls coalitions of divergent interests.

Competition and divergence is, so to speak, embedded into the economic and structural foundation of the relationship whereas collaboration is a more ephemeral and vulnerable dimension that is embedded into the culture of the relationship. McConnel suggests that to build and maintain a well-functioning collaboration both parties must respect the other part’s “Bill of Rights” [7]. Customers must, for example, formulate the goals and expect to have them followed, and they have the right to make reasonable requirements changes and to know their price. Developers must, correspondingly, know the goals and the underlying priorities, and they must be able to finish each activity in a technically acceptable fashion. The challenge is, in more general terms, to build and maintain a collaborative environment and culture between the customer and the supplier in which the underlying competitive nature of the relationship is explicitly addressed and dealt with.

Trust and Control

At the contractual level we find a psychological contract as well as a formal, written one [10]. The psychological contract builds on trust and consists of a set of unwritten and
largely unspoken congruent expectations. The written contract offers mechanisms for structural control and is the outcome of an explicit negotiation process. Sabherwal [10] argues that trust—meaning “confidence that the behavior of another will conform to one’s expectations and in the goodwill of another”—improves performance and distrust hurts performance in IT projects. Trust is embedded into teams of collaboration whereas structural control is an expression of the bureaucratic and market dimensions of the relationship. Both structural control and trust is needed to address unexpected problems, and projects must constantly seek to balance the two to effectively manage the uncertainties and risks they will face.

Structural control is important to make customers confident in a project’s outcome and suppliers confident that the users will not ignore the project or unceasingly demand new requirements. Too much control can lead to too much time spent on reporting progress instead of developing software, and too little control may lead to inefficient, or lack of, communication and co-ordination which in turn results in distrust. Too much trust indicates that one party is giving away their responsibility, and too little trust means that communication becomes increasingly difficult.

Sabherwal [10] distinguishes between four types of trust:
1. Calculus based trust is rooted in rewards and punishment in the project. This type of trust can be established through structural control and penalty clauses to minimize opportunistic behavior, through project management and reporting procedures, and through the customer’s recognition of the supplier’s desire to contract future projects.
2. Knowledge based trust depends on the two parties knowing each other well. It can be brought about by experience from earlier projects, by asking the supplier to demonstrate its ability either by running a test case (for standard software) or by building a small system first, by key persons in the organizations knowing each other, or by “courtship” where the partners seek to know each other well before starting the project.
3. Identification based trust is an expression of the parties identifying with each other’s goals. It is enhanced by establishing shared goals and mutual understanding through early team building efforts or through collaboration in analysis and design.
4. Performance based trust is based on project successes. It is especially important when the customer and the supplier are geographically distant. It can be supported through rituals when intermediate goals are reached or through demonstration of completed portions of the system.

Sabherwal [10] argues that well-functioning customer-supplier relations must balance trust and structural control. Structural control is needed to promote decisions and monitor progress, and trust is needed to promote creativity and mutual learning.

**Care and Engineering**

The development level of the relationship is where things get done. A technical system has to be developed or adapted to the customer organization and this task involves a number of software engineering activities. A strong focus on engineering is both natural and needed. But when it is combined with a neglect of the problems and challenges related to the organizational implementation of the new system it becomes highly
problematic from the customer’s point of view. Ciborra [2] suggests that when an organization introduces a new IT-system, it needs to practice hospitality, i.e. it needs to be open and friendly towards the new “guest”. Hospitality is necessary due to the ambiguous nature of the system. The system can turn out to become a good friend, supporting work in the customer organization in a flexible and efficient manner. Or, it can turn out to be a monster, making everyday life difficult and troublesome for the users. In the latter case, hospitality easily turns into hostility. Being hospitable means to take care of the new system at three different levels.

Perception is the kind of care we show to anticipate the new system. To care project teams from both parties develop shared visions of the new system, they build prototypes, expectations to the new systems are adjusted, change control is applied, acceptance test is well planned, and user training is emphasized from an early stage of development.

Circumspection is the kind of care shown to make the system easy to use. It includes error prevention by training users and operators, ad hoc error correction, collecting change orders, and changing routines so that the fit between system and work tasks in the customer organization is good. A help desk function is also a typical example of this kind of care.

Understanding, the third level of care, is reached when the new system is integrated as a well-functioning part of the customer organization. At the circumspection level the system is present-at-hand [13] and it causes breakdowns in the customer organization. In the understanding phase the system becomes ready-at-hand [13], i.e. the use of the system is now an integrated part of the primary work task, and there is no longer a need to focus on the system as such.

The challenge in systems development is to build and implement a new or and adapted IT system. Construction of a technical system requires that engineering disciplines be applied to perceived or stated requirements. If the development process is dominated by engineering concerns there is a considerable risk that the business impact of the effort will be rather weak in the customer organization. Care is needed to deal effectively with the social complexity involved in IT implementation.

**Relationship Dynamics**

The presented pairs of concepts are expressions of the contradictory nature of the relationship between customers and suppliers in IT development. They represent ‘soft’ values related to collaboration, trust, and care. These values are expressions of a romantic worldview [3] focused on mutual learning, creativity, and change. But this romantic position is complemented with ‘hard’ values related to competition, control, and engineering. These values are expressions of a mechanistic worldview [3] in which the emphasis is on prediction, method, and order. There is a tendency in out-sourced IT development to focus on competition, structural control, and engineering, and to enforce these values in face of uncertainty or when unexpected events emerge. What is needed is, instead, to create a balance in which the contradictory relation between customers
and suppliers and between present and emerging organizational forms is explicitly dealt with [9].

The three levels of contrasting opposites are intrinsically related. If a customer and a supplier, starting from the constituting level, have a relationship based on collaboration and acknowledge a common goal, this often encourages trust. When unexpected events occur, as they invariably do, this normally results in negotiations. In an atmosphere of trust, it is easier for each party to go for a win-win solution, and to believe that the counterpart does the same. Trust furthermore gives room for learning, for example to experiment in face of uncertainty or to change decisions when emerging and unexpected insights call for it. In a similar fashion there is a tight interaction from the lowest level upwards. When projects are based on care, it reinforces trust at the contractual level and this in turn enhances and further develops collaboration at the constituting level. The three levels are also connected by the other opposites of the pairs. Competition results in the enforcement of structural controls that are used for controlling the other party, either that the supplier delivers software as promised (thus neglecting the implementation process) or that the customer delivers involvement as promised.

Even in a well-functioning process that dynamically addresses and manages the relation between collaboration and competition, trust and control, and care and engineering, problems and unexpected events do arise both due to the complexity and uncertainty of the task and also due to the inherent, social complexity of such collaborations. A balanced approach is in no way a guarantee for success. But it does reduce the risk of failure or unsatisfactory outcomes.

**Maturity Models**

Recent maturity models address the customer-supplier relation in IT development. We will review two specific models, SA-CMM and SPIRE, in the light of the concepts above. The models are briefly described in the separate textbox. Their contents overlap, even though their structure differs.

Both models support handling the competitive nature of the relationship. Competition between suppliers is taken care of, e.g., by using clear evaluation criteria when selecting a supplier. Competition between the customer and the supplier is taken care of through cautious choice of contract type. The models are not specific on penalties and rewards as this is related to the chosen type of contract. Nevertheless both models suppose some kind of fixed delivery date and fixed price. Both models also support structural control. They emphasize that plans are important, that it is necessary with change control procedures, and that the contract, including the requirements, should be tracked and monitored. Finally, the models support engineering. They emphasize that the quality of the delivered product is tested and evaluated. This is done through the acceptance test, but peer reviews of intermediate products are also included. As a consequence, both models have, as indicated in Table 1, a high score on mechanistic aspects of the customer-supplier relation. They do, however, differ when it comes to the romantic criteria for a well-functioning relationship.
SA-CMM

The Software Acquisition Capability Maturity Model (SA-CMM) is built in the same manner as the SE-CMM (Software Engineering CMM) and described in [4]. An organization can improve itself by improving the defined key process areas at each of five maturity levels. The focus is, as the name says, on the process of acquiring a system, not on that of engineering one. The organizations for which SA-CMM is relevant will typically be large organizations with separate purchasing departments. SA-CMM is oriented towards organizations that acquire systems fairly often, and it helps them match suppliers that use CMM [5]. This is especially relevant for suppliers to military organizations, because they are required to perform at CMM level 3 with sound project management practices and organizational routines that are used by all projects.

SA-CMM is described in the same manner as CMM. Each level has a set of key process areas. For each of these, the following is described: goal, commitment to perform, ability to perform, activities performed, measurement and analysis, and verifying implementation. SA-CMM level 2 is most directly concerned with the collaboration with the supplier. This level has seven key process areas: software acquisition planning, solicitation, requirements development and management, project management, contract tracking and oversight, evaluation, and transition to support.

SPIRE

The SPIRE handbook [11] is for small software development companies (less than 50 developers) on how to conduct software process improvement. One chapter (chapter 8) is about the customer-supplier process and it focuses on several processes that may be improved by a supplier. The customer-supplier process is described as follows: 1. Acquisition process, 2. Supply process, 3. Requirements elicitation process, 4. Operational process.

Each process is described in two sections, Process Basics and Implementation Guidance. Process Basics contains the following subsections: purpose, motivation, inputs, activities, outputs, industry best practice, and results expected. Implementation Guidance contains the following subsections: information, things to do, things to avoid, things to think about, and further reading.

SA-CMM focuses on management and control of the software acquisition process. It recognizes different affected groups, and the need for communication among these. But collaboration and trust are not seen as important aspects of the process. The model does for example not emphasize collaboration as a necessary means to reach the goal, and there are no indications of how the customer and supplier come to share goals and success criteria. Trust is only included through structural control and the psychological contract is not mentioned. Care in the form of perception is shown in that affected groups are involved in requirements development and management. Care in the form of circumspection is shown in a “transition to support” process that makes sure that there is a support organization, at least at the operational level, that can correct errors, make
changes, and take responsibility for day-to-day operation. But being a model for software acquisition, we expect more about user training, the fit between procedures and the system, and the interface between the users and the supported system, including user documentation, and support facilities like help desk. In summary, it seems as if the target group for SA-CMM is the purchasing part, not the user part of the customer organization.

SPIRE is more concerned about collaboration, trust, and care. This is mainly visible in the Implementation Guidance sections. It is taken for given that the customer and a supplier have a shared goal, a successfully implemented system. It is also taken for granted that the customer and supplier organization will perform joint activities. Nevertheless, areas where disagreement often occurs are dealt with explicitly, like the need to agree on requirements, to agree on the boundaries for what the customer and supplier organization are responsible for, and on whether the system is acceptable or not. SPIRE is fairly direct about trust: “...suppliers must be able to deal effectively and appropriately with their customers in order to deliver their products and services predictably and profitably and in the first place give them confidence that this can be achieved” (p.75).

<table>
<thead>
<tr>
<th>Worldview</th>
<th>Aspect of relationship</th>
<th>SA-CMM</th>
<th>SPIRE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mechanistic</td>
<td>Competition</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td></td>
<td>Structural control</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td></td>
<td>Engineering</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Romantic</td>
<td>Collaboration</td>
<td>*</td>
<td>*****</td>
</tr>
<tr>
<td></td>
<td>Trust</td>
<td>*</td>
<td>****</td>
</tr>
<tr>
<td></td>
<td>Care</td>
<td>***</td>
<td>****</td>
</tr>
</tbody>
</table>

Table 1. Evaluation of models (*: poor; **: a little; ***: mediocre; ****: good; *****: very good)

SPIRE proposes different means for promoting trust. Calculus based trust is secured by structural control, but also in recommendations like “Ensure that there is an agreed procedure for contact between you and your customer” (p. 94), and “Understand customer expectations” (p. 92) which support the psychological contract. SPIRE also proposes activities for increasing knowledge based trust, like obtaining details of other projects related to the required system, investigating turn over of staff along with their skills, qualifications and experience, and looking closer at the supplier organization’s commitment to software improvement and quality assurance.

In addition, SPIRE is focused on care. Perception is promoted by emphasizing the needs to be met by the system and the systems significance and value to the business. Real users should also validate the requirements, and key users should be involved in the initial analysis of needs. Circumspection is supported by advocating appropriate user training, adequate user documentation, a procedure for handling user requests, and through provision of assistance and consultation to the users on the new software. Thus SPIRE gets full score on collaboration, and nearly full score on trust and care. Concerning trust, “courtship” is missing, as well as performance-based trust.
Concerning care, we miss the fit between organizational procedures and the use of the system.

In summary, we evaluate SA-CMM as a mediocre model for acquiring systems, seen from a customer point of view. It does not propose a useful balance of mechanistic and romantic aspects of the relationship, and it disregards the activities needed to implement the new system into the customer’s business processes. We evaluate SPIRE as being a good model for system acquisition because it balances mechanistic and romantic criteria fairly well. It does not get a full score, however, because the model can still be improved through an explicit treatment of the relation between mechanistic and romantic criteria.

The Workshop

SSE’s management signals a strong will for collaboration. They see collaboration with the customer as a competitive advantage and they want to improve the relationships. SSE therefore decided to organize a workshop to initiate collaboration with customers, to evaluate strengths and weaknesses in the existing relationships, and to identify possible actions that could be initiated to mature collaboration. The workshop was designed as a mixture of prepared presentations and structured joint discussions. A number of customers were asked to present assessments of the collaboration, and we presented state-of-the-art knowledge on customer-supplier relations in IT projects. In the following we evaluate the customer-SSE relationships based on the strong and weak sides that emerged at the workshop and framed by the theory presented above.

The seminar itself is an expression of a co-operative strategy, but SSE management demonstrates it more clearly in some projects by taking on unexpected costs in order to provide satisfactory solutions to the customer. The customers emphasize SSE’s will to co-operate and there seems to be a useful balance between competition and collaboration at the constituting level.

The customers also mention trust as a key part of their relationship with SSE. Identity-based trust, i.e. the will to identify with each others’ goals, is very much supported by SSE management, but it needs to be strengthened at the project level as expressed in comments like “lack of sharing knowledge” and “one-sided dependency”. The amount of knowledge-based trust varies with the customers. Some relationships are old while others are fairly new. Nevertheless, lack of knowledge about the customers’ organization is mentioned as a common problem. In general there is trust in SSE’s knowledge. This can be seen in positive comments like “good professional knowledge”, “good technical knowledge”, and “good competence”. These statements indicate that most of the time, the customers are satisfied with SSE’s performance. There is a certain level of performance-based trust. But statements like “unpredictable delivery time” and “lack of domain knowledge” indicate that there is room for improvement. The most pressing problems relate to calculus-based trust, however. Even though calculus based trust is supported through penalties in the contracts, there is lack of structural control at both SSE’s and the customers’ side. The lack of structural control at SSE’s side is expressed as “unclear expectations”, “lack of control”, and “unpredictable delivery time”. The customers also express a wish for formalized contact and communication
during the whole project and they would like more structural control, for example in
terms of better project organization.

SSE is a software engineering company that only to a limited extent is involved in the
perception aspect of care because projects find it difficult to get access to “real users”
(most customers participate at a mediating level where contracts are made). One
important contribution to perception, as seen from the customers’ point of view, is to
critically question the requirements, to avoid the expensive, unnecessary ones. To a
certain extent, SSE does not support circumspection in that the documentation could be
better. At the seminar, customer representatives also said that they themselves are too
little concerned about care, both with respect to perception and circumspection. It is
difficult to get the right users to participate in the projects, and too little care is taken to
motivate and teach new users, and to give them proper support.

There are, in summary, important unbalances between trust and control and between
care and engineering in the relation between SSE and its customers. The lack of
structural control, the possibilities for distrust implied by this, the strong tradition for
focusing on software engineering and traditional software process improvement, and the
relatively weak attention, on both sides, paid to care, are only to some extent
counterbalanced by SSE management’s general focus on collaboration. There are,
indeed, many good reasons to improve the relationship between SSE and its customers.

Below, in Table 2, we list the proposals for improvement that are forwarded at the
workshop. The participants divide the proposals into customer’s, shared, and supplier’s
improvement. The authors have done the additional grouping. Even though some of the
proposals encourage stronger structural control, like the suggestions under project
management and requirement management, the focus on improved knowledge about
and understanding of the other party is striking. The suggestions on contextual issues,
mutual understanding, and process support all point in this direction, as well as the
suggestions under organizational implementation and partly under miscellaneous. The
focus on mutual understanding, and the communication needed to achieve this, is a key
aspect of the romantic worldview. This indicates that if SSE wants to improve the
relationship to its customers, it must expand the romantic aspects at the project level and
more systematically balance this with mechanisms for structural control.

**Early Initiatives**

Based on the workshop SSE has initiated specific improvement initiatives targeting the
customer-supplier relationship. First, a number of joint workshops are being designed
and implemented in which customers can take active part. Second, a number of
customer initiatives will be supported by SSE services to strengthen the customer's
participation in projects. Examples of such initiatives are development of project
management skills, establishment of a project culture, and opportunities for SSE
developers to visit and interview “real user”. And third, SSE will develop simple
frameworks, partly based on the available models, which can be used to critically assess
the customer-supplier relationship. SSE will take the initiative and customers are invited
to participate in joint efforts.
Table 2. Proposals for improving the customer relation at SSE

As a first step, SSE has developed a framework for improving customer relations including six workshops:

- **Project establishment.** This workshop establishes the project’s vision, it focuses on goals and the various roles of customer and supplier, and it includes a risk analysis session.
- **Establishing requirements.** This workshop involves relevant stakeholders in identifying key requirements and in deciding on procedures for managing requirements changes.
- **Project diagnosis.** This workshop starts out with a review of the project’s schedules and budgets and continues with an evaluation of the results. Again, a risk analysis session is included.
• Rollout. This workshop focuses on the organization that is going to use the new system. A rollout strategy is defined, the various roles involved are identified, education and training needs are analyzed, and the implementation process is planned.
• Test planning. This workshop emphasizes test strategy, design of acceptance criteria, development of test plans for customer as well as supplier, and provision of test data.
• Project evaluation. This workshop analyses positive and negative experiences and develops proposals for how to improve future co-operation.

A workshop
A project evaluation workshop was held with a customer that has several years of experience with large SSE projects. The workshop was carried out as a structured brainstorm focusing on why the project went well and how it could have been better. Comments were grouped and given subheadings to capture the essence. The subheadings were then presented and discussed. The project went well because:
• A “fantastic partnership” with mutual goals, respect and trust led to a flexible approach that ensured the success of the project.
• Meetings were well organized with clear agendas and constructive discussions.
• Management meetings and technical meetings were separated.
• There was contact at all levels between the parties involved resulting in direct access to knowledge, information, and decision-makers.
• The required technical competence was present.
• The contract supported that all parties shared goals and were motivated to complete the project within time and budget.

A number of possible improvements were also identified:
• Control of subcontractors could have been better.
• It was sometimes difficult for SSE to obtain the information needed because other customer activities were given higher priority.
• SSE lacked operational knowledge from the future users, and was therefore often dependent on the customer.
• Top management at both sides could have been more involved in the project.
• The contract may be simplified by removing those milestones that are not needed from a practical point of view.

SSE has subsequently identified five recommendations for future projects:
• Define a strategy for end user involvement.
• Aim at common goals for the end users, the customer, and the contractor.
• Establish communication and contact at all levels in the involved organizations.
• Define only practically useful milestones.
• Consider user documentation and training up front.
Conclusion

There is a gap between what we know about IT projects that crosses organizational boundaries and contemporary models for improving customer-supplier relationships. The SA-CMM has a strong orientation towards competition, control, and engineering, while the SPIRE model takes a somewhat more balanced approach. Both models can, however, be improved by systematically applying state-of-the-art knowledge on systems development to include mechanisms and concepts that facilitate collaboration, trust, and care.

Generally we agree with the CMM that any IT organization must start by improving its project management practice. It makes little sense for a software organization to invite customers to joint improvement initiatives before its own project management disciplines are in place. Software development is, however, a complex exchange process that crosses organizational boundaries. As software organizations reach level 2 they therefore need to go beyond a traditional focus on internal processes and include customer relations into their SPI programs.

Too few SPI initiatives take this position serious. Our diagnosis of SSE customer relations reveals strengths, weaknesses, and many opportunities for action. Our experiences from this effort support the idea that joint improvement efforts should lead to a useful balance between mechanistic and romantic aspects at all levels of the relationship.

References


